Simplify Mobile Phone testing

by Randy Kemp and Roger Richardson

Abstract

Today companies need a solid method for testing smartphone firmware, application upgrades, software upgrades and new releases.

New features are constantly introduced into smartphones. Interactions between software upgrades, firmware, software and user interactions are increasing in complexity. New phone models are constantly introduced into the marketplace. Users keep demanding new features and are given more marketplace choices. The demand on testing teams is stressful.

Inadequate testing is a constant challenge. The world population has an increasing demand for smartphones. It's estimated that about three-quarters of the US population will own one by 2013. Users prefer to use smartphone cameras over their own cameras. The phone system API encourages third-party application development. A constant demand for new features, phone models and applications, increases the risks of testing errors.

Testing teams must have a systematic smartphone testing approach. Users must adapt an approach that looks at three testing categories:

- Phone states: There are dynamic states to consider, where the software actively works on data. There are static states, where the user is left at a specific place for functional control.

- External interruptions: Interruptions beyond a users control

- Orientation: Adjustment of the physical positioning of the device, that can affect both the display and change functional actions

This paper will introduce you to a testing method, designed to cut user calls to call centers, test the largest smartphone possible scenarios, and increase testing team success.

© 2012 by Roger Richardson and Randy Kemp

Mobile phone Complexity and testing: The big picture

Brief history

There are two types of phones testers need to test, according to Phone Scoop

- Feature phones – According to Wiki, it's a " mobile phone which at the time of manufacture is not considered to be a smartphone, but has added functions over and above standard mobile services. It is intended for customers who want a lower-price phone without all the features of a smartphone." In addition, it is operated by proprietary firmware.

- Smartphones – According to Wiki, "A smartphone is a mobile phone built on a mobile operating system, with more advanced computing capability and connectivity than a feature phone."

Feature phones are still popular

Testing teams must still focus on feature phones. These lower cost devices are popular with senior citizens and other groups.

A comprehensive testing approach should cover both. According to Don Kellogg,"in 2011, feature phones accounted for 60 percent of the mobile telephones in the United States and 70 percent of mobile phones sold worldwide."

Yet smartphones are increasing in popularity each year:

- venturebeat.com conducted a survey in 2012. According to the survey, about 50% of US mobleconsumers owned smartphones. By 2013 this number would increase to 70%.

- According to telecomsmarketresearch.com, the number of active European mobile subscribers is 860 million.

A good sign of increasing mobile phone usage, is the number of application downloads. The mobile phone applications continue to multiply. Let's look at some data from a September 2012 research report by Gartner:

© 2012 by Roger Richardson and Randy Kemp

• Free Mobile applications make up 89% of all downloads – most others are under $3..

• They estimate in 2012 there will be 46 billion mobile application downloads. This is almost twice the 2011 figures of 25 billion downloads.

• They are predicting by 2016, there will be 310 billion app downloads – 93% being free apps.

Downloads add complexity

They also increase consumer demand for mobile phones. Increased demand brings increased testing needs.

Let's look at a Table Gardner supplied:

Mobile Application store downloads – Worldwide (millions of downloads), 2011-2016

Mobile Application store downloads – Worldwide (millions of downloads), 2011-2016						
	2011	2012	2013	2014	2015	2016
Free downloads	22044	40599	73280	119842	188946	287933
Paid for downloads	2893	5018	8142	11853	16430	21672
Total downloads	24936	45617	81422	131695	205376	309606
Free downloads %	88.40%	89.00%	90.00%	91.00%	92.00%	93.00%

During September 2012, Hugo Barra, Google's Android Director of Product Management , shared some interesting data on Android. He said this:

• 500 million Android devices are globally active

• Each day users add 1.3 million Android devices

Operating systems introduce complexity

To keep up with demands, manufacturers created various smartphone operating systems. The best know are IOS from Apple and Android from Google. Microsoft attempts to partner with manufacturers to gain market share.

© 2012 by Roger Richardson and Randy Kemp

Operating systems introduce complexity

To keep up with demands, manufacturers created various smartphone operating systems. The best know are IOS from Apple and Android from Google. Microsoft attempts to partner with manufacturers to gain market share.

Just as PC operating systems can multi-task, so can smartphone operating systems. This means testers need to look at combinations - not just serial events. Can we capture various multi-task operations

User Interaction

One thing missing from many testing approaches is user interaction. Perhaps the user is downloading email, running a weather application, checking on stock picks and looking at his calendar. These user tasks are in an active state. The user can interrupt these tasks or start new ones. Do current testing approaches take user interaction into account?

Are smartphones easy to use?

It appears so on the surface. But underneath with all the software and hardware, many surprises can challenge your testing team.

Consider testing a new smartphone. New applications in place must work with old software and co-exist with a user's expected behavior. We're not just testing a simple phone. Now we're testing smart phones with more capabilities and possibilities to include. What happens if the user changes a phone's orientation? The user could change the physical phone's position. This can effect both the display and change functional approaches.

Here's an example scenario to look at: the application must work properly with all user actions. Possible user actions are button , screen presses and orientation adjustments.

Challenges testers face

In 2011, IDC did a comprehensive study of software developers. One study aspect focused on the top challenges they faced, when developing and testing mobile application. Here are the challenges and percentages of respondents:

© 2012 by Roger Richardson and Randy Kemp

Challenges and percentages of respondents

Challenges	Approximate percentage of respondents
Managing many different device targets	60
Lack of consistent standards	48
Resource constraints of mobile devices	45
Managing of different operating system environments	37
Managing special API versions	33
Finding and integrating appropriate development tools	25
Insufficient development methodologies	23

Tip: Testers encounter new challenges - continued

Challenges	Approximate percentage of respondents
Developing security for web services clients	20
Other	7
None/don't know	2

Looking at combination complexity

• How would you approach testing the following variables?

• Suppose you had forty applications - twenty to fifty possible user interactions.

• Interruptions from users (includes all screen actions, camera and buttons, plus reaction to user actions) - about one hundred possibilities.

• Interruptions from external events beyond the users control (i.e. network changes WIFI, 3G and GSM, incoming audio/ video calls, Emails, MMS, etc.).- about sixty possibilities.

• Interruptions from accessories, like external power and Bluetooth - about 20 possible.

© 2012 by Roger Richardson and Randy Kemp

• Orientation adjustment with six possible orientations per single screen

• Orientation adjustment with six possible orientations, with both screens open - rotated right 90*, left 90*, normal, 180 degrees and face up/down.

• If we considering both the starting and ending orientations in the two previous orientation adjusts, it means each test case has 144 possible orientation combinations.

Tip: Don't be overwhelmed

The actual number of test can interactions might surprise you. But a solid testing approach will lead testing teams down the right road.

Total combinations
Applications: 800 to 2000 states
Interruptions: 100 + 25 + 60 for 185 events
Orientations : 25
The low grand total is 800 * 180 * 25 for 3,600,000 possible 2 level test cases
The high grand total is 2,000 * 180 * 25 for 9,000,000 possible 2 level test cases

This is a staggering number to ponder.

An approach to testing

Test case designers try to explore all testing possibilities. But thinking of possible interactions is difficult. Such an approach requires a structure. For practical purposes, we will include software applications (.ie GPS) under the feature umbrella.

Next we will look at some test case categories. The suggested categories are static states, dynamic states, external events, user actions, actions and accessories.

© 2012 by Roger Richardson and Randy Kemp

Actions	Definitions
Accessories	Accessories refer to devices that can function, without necessarily being connected to the device. Physical or radio methods may provide the connection link. Some examples are Bluetooth connections, charger connect/disconnect and wired headset connect/disconnect.
External events	External Events are anything beyond the user's control. The user does not have to start any item. They can occur without user intervention. When the user starts them, they become user events. Some examples are incoming IM, incoming SMS, alarm outputs or automatic network switching (GSM/WIFI/3G/LTE).
User actions	User actions refer to anything the user can start, while the present state exists. All buttons and applications are open, as the source for these actions. Examples are volume adjustments and pushing camera button.

State types	Definitions
Dynamic states	Dynamic states represent conditions that are doing an active process, like saving a file, browsing the web or playing a game. The interruption has a possibility to interfere with its operation. Some examples are playing a video or audio, having a browser web site updating, or downloading a video/audio file.
Static states	State definitions are taken from the requirements for the application. They are divided into user attainable segments. Static states show the user at a selectable prompt or entry - like selecting a video to play. Some examples of states are selecting a video application, selecting to play a video, having the video selection list shown or having a video chosen (but the run button not selected).

Tip: Test effectively

© 2012 by Roger Richardson and Randy Kemp

Now we will introduce a method called Feature Interaction Matrix. It gives testers and management a way of perceiving system level and behavioral issues not seen by normal testing methods.

By arranging the data into rows (states) and columns (Events), multiple interactions can be observed. Test cases can be selected, where row and column met. Testers making test cases no longer have to rely on their memory. This approach will choose interactions to work with.

Others can accumulate the events and add to the tool. You can create this method with a simple Excel sheet or an online system build in platforms like .NET, PHP, Mysql and PostegeSql. We will illustrate this system though a series of diagrams.

> **Tip: Let FIM find what you can't**
>
> It's hard for the human mind to think of all possibilities. But a good testing methodl can do this easily.

What it solves

Test case selection becomes a new issue. There are hundreds of combinations to select from. Just choosing the right combination to find issues, can bury the tester. It's hard to restrict selecting test cases, when they are so easily created. Are matched pairs an answer? Some pairs are more interesting than others.

Data interruptions

When a data interruption interferes with a data action, the program controls it. The programmer is quite ready to solve those issues. Crossing the functional edge of the control structure, leaves the system with different considerations.

Audio hand offs

Audio hand offs are particularly sensitive and may find some very unusual issues. Some examples of incomplete hand offs are:

- Muting that reaches into areas where they shouldn't
- Volume adjustments that are either maintained or lost when switching sources

© 2012 by Roger Richardson and Randy Kemp

• Audio control that gets locked with the interruptions control and never returns to system control.

Network switch overs

Network switch overs happen in the background, but can be of real interest. This is especially true when multiple RF sources switch back and forth. Consider concentrations on doing the possible RF switching between GSM, LTE, 3G,WIFI and Bluetooth. Concurrency issues can surface, where each action and program by itself works fine. When we join them together, it becomes inconsistent.

Cross audio control and data transfer items

Cross audio control and data transfer items are a great source of issues. Your ears can pick up a single missed note or slur in the replay, while data transfers area active. Look at it this way: data interruptions are used to handling the data control. When the effect of audio comes into the play, the data control might take too much processing power. It could drop audio or suspend the audio actions. Users are very sensitive to missing the beat. So these types of transitions are a primary interest.

Tip: Save money with sound testing

Architectural visualization

User initiated background applications - like a music player - adds another diminution to testing. T Picture the music player in the background, with another non-audio application, working in the foreground. Then put an incoming call in place. You would think that it is the same thing as testing the music player in the foreground, with an interruption - but it isn't. Problems found can include:

- Loss of control of the audio
- A complete lack of being able to return to the background player.

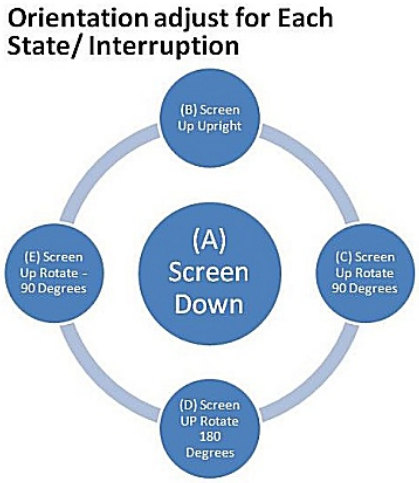
© 2012 by Roger Richardson and Randy Kemp

This same style of testing should also extend to streaming audio on a browser if it can play in the background.

Using the method

Let's explore the method through some visual displays.

Orientation adjustments for each state/interruption.



Tip: Automation is key!

Testing this manually would be quite a chore! Imagine how much easier this is using automation?

25 possible combinations for each orientation

S1/IE1 Orientations	(A) Screen Down	(B) Screen Up	(C) Screen Up Rotate 90 Degrees	D) Screen Up Rotate 180 Degrees	E) Screen Up Rotate -90 Degrees
(A) Screen Down	A/A	A/B	A/C	A/D	A/E
(B) Screen Up Upright	B/A	B/B	B/C	B/D	B/E
(C) Screen Up Rotate 90 Degrees	C/A	C/B	C/C	C/D	C/E
D) Screen Up Rotate 180 Degrees	D/A	D/B	D/C	D/D	D/E
E) Screen Up Rotate -90 Degrees	E/A	E/B	E/C	E/D	E/E

© 2012 by Roger Richardson and Randy Kemp

Matrix Layout

This diagram shows how to select an individual test case from the matrix. The intersection of S1 and EI1 creates a test case. The user starts with an S1 state, at a selected orientation. It is adjusted to the interruption orientation, and then the EI1 interruption occurs.

State types	Definitions
Dynamic states	Dynamic states represent conditions that are doing an active process, like saving a file, browsing the web or playing a game. The interruption has a possibility to interfere with its operation. Some examples are playing a video or audio, having a browser web site updating, or downloading a video/audio file.
Static states	State definitions are taken from the requirements for the application. They are divided into user attainable segments. Static states show the user at a selectable prompt or entry - like selecting a video to play. Some examples of states are selecting a video application, selecting to play a video, having the video selection list shown or having a video chosen (but the run button not selected).

Benefits

We hope you see how this automated approach is superior to current manual approaches. Automation of writing the test cases saves immense amounts of time. Typical text case output can easily be 200 – 300 test cases - per tester -per day. More test cases can be written than there is time to test. The problem is now making the correct test case selections.

Tip: Have the right smartphone testing compass

The benefits of this approach are immense.

- Because more test case interactions can be accounted for, there should be a significant reduction in phones being returned.
- The tester is free to focus on other tasks, since the combinations are automatically calculated.
- Management and marketing can focus on new mobile phone releases, without sacrificing quality.

© 2012 by Roger Richardson and Randy Kemp

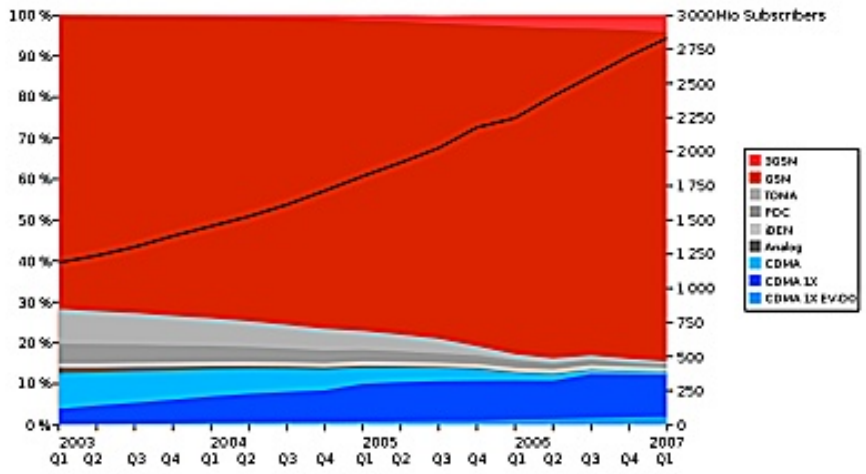
• Users are happier with the phone products, as they should have a better communication experience with others.

Why wait?

Mobiel usage keeps increasing each year, as the following graph illustrates:

> **Tip: Take the next step**
>
> Contact the authors. Find out what FIM can do for you smartphone testing.

Cellphone subscribers by technology



Wouldn't you like to have a sound testing methodology in place?

© 2012 by Roger Richardson and Randy Kemp

About the Authors

For more information on the FIM test case approach, please contact the authors at their given website and/or social media outlets.

Roger Richardson spend several years doing US Navy computer and hardware repairs. He then spend several years working with Motorola Mobility – now owned by Google - on leading cell phone technology. Motorola granted him six patents for ideas he submitted. He's available as a technology consultant, as well as an expert on various web-based programming languages. You can find out more about him at http://tinyurl.com/rr-il-linkedin.

Randy Kemp spent 30 years as an employee and consultant for several software environments, like Motorola, Allstate, Discover Card and United Airlines. He's now a copywriter, writer and ghostwriter, who specializes in software, direct response and viral marketing. Randy is a Motorola trained, six sigma black belt. He spent several years developing software for Motorola Mobility, now owned by Google. There he had a chance to work with Roger on his test case method. You can find out more about him at http://tinyurl.com/rlk-googleplus.

© 2012 by Roger Richardson and Randy Kemp